PASSED

Group 3 - lab 5

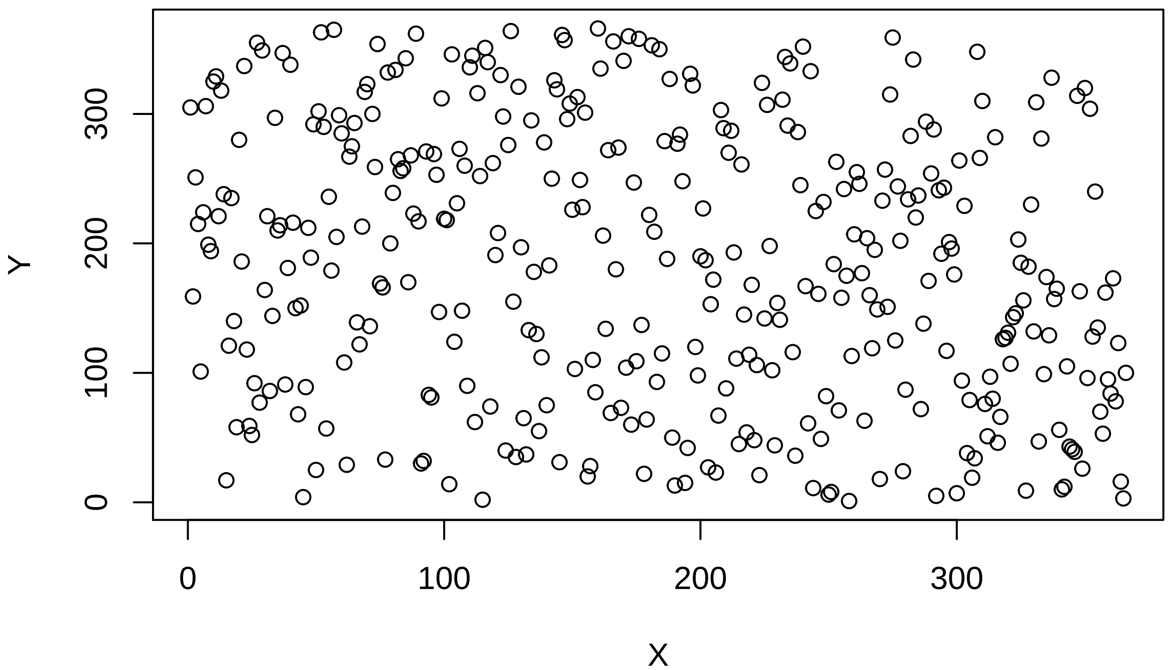
Nicla Lovsjö & Maxime Bonneau

10 march 2016

Assignment 1:

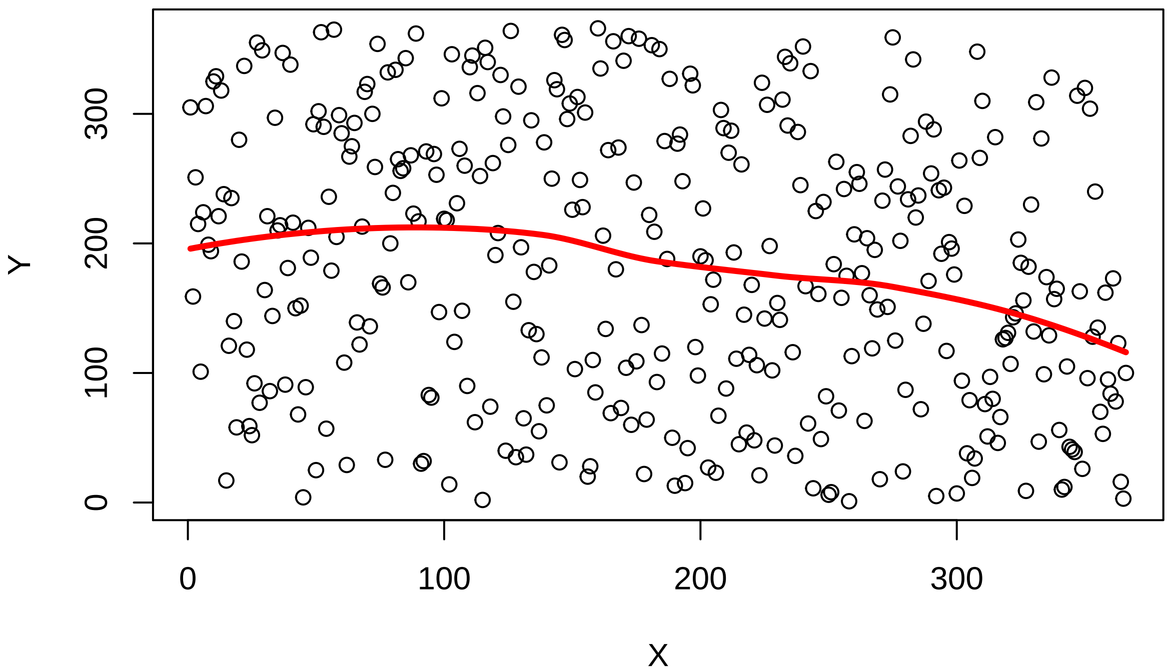
1.1

We read in data and make a scatterplot. From this plot alone, we can not say that this is non-random.



1.2

We make a LOESS-model of Y X and get an estimated $\hat{Y}$. We put the curve $\hat{Y}$ into our original plot.



From this we can start to suspect that this is not random, since we have the decay in the right-hand-half of the plot. We still dont know if this decay is enough to claim that this is non-random though.

1.3

We use the statistic,

$$T = \frac{\hat{Y}(X_b) - \hat{Y}(X_a)}{X_b - X_a} \tag{1}$$

We estimate the distribution of T by using a non-parametric bootstrap. From T and the plot above we see that IF some data is non-random, it would produce a shape of $\hat{Y}$ that is different from a straigth line. It would not matter if it goes high to low or low to high, i.e. if abs(T) is significantly different from zero. The mean of our sample is -0.35, and our null-hypothesis is that Lottery is random, i.e. T is equal to zero. Now if we bootstrap our sample and count the number of bootstrap-values that are over 0, we can reject H_0 if percentage of such values is more than, say, 0.05. We get a p-value of 0.002 so here we can reject the null-hypothesis and conclude that the data is random.

```
## [1] 0.002
```

```
## [1] 0.0015
```

1.4

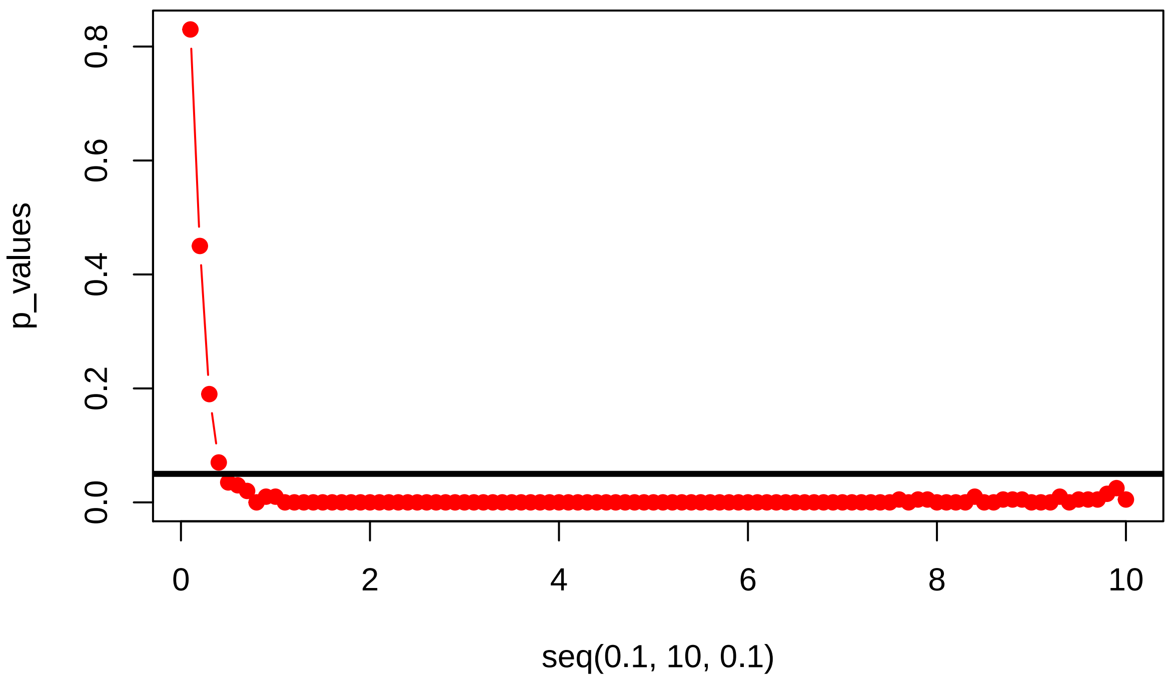
Now we test the same hypothesis using the permutation test. That is, we permute our target variable and if it is random this will not matter, such that the observed T-value will not be extreme in any way, i.e. in some rejection region looking at the distribution.

```
## The p-value for perm.test is: 0.143
```

Here we can actually not reject the null-hypothesis. This is strange, since we then are saying different statement about the randomness in the data using two different methods. We dont know why this is.

1.5

Lastly we look at the power of the tests. We construct a loop that uses the given α-values, which gives the following plot:

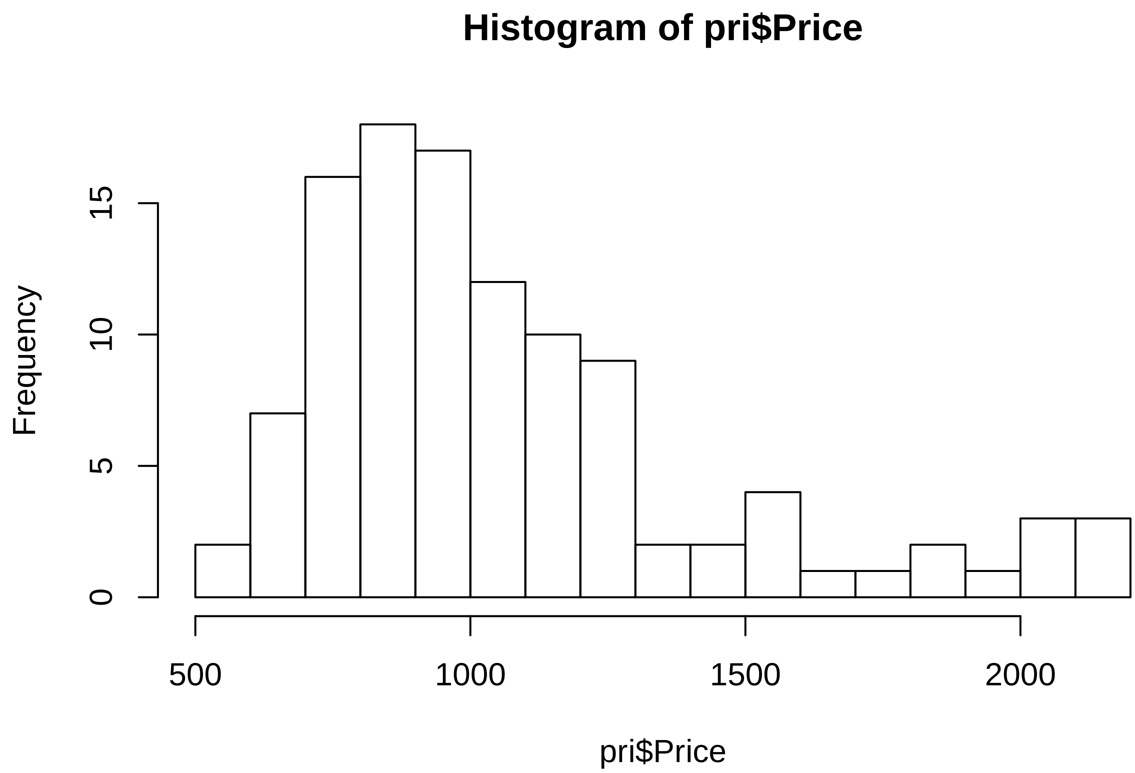


So we see that for the low α-values we can not reject the nullhypothesis but as α gets bigger the data becomes less random, and the permutation test is able to find this result.

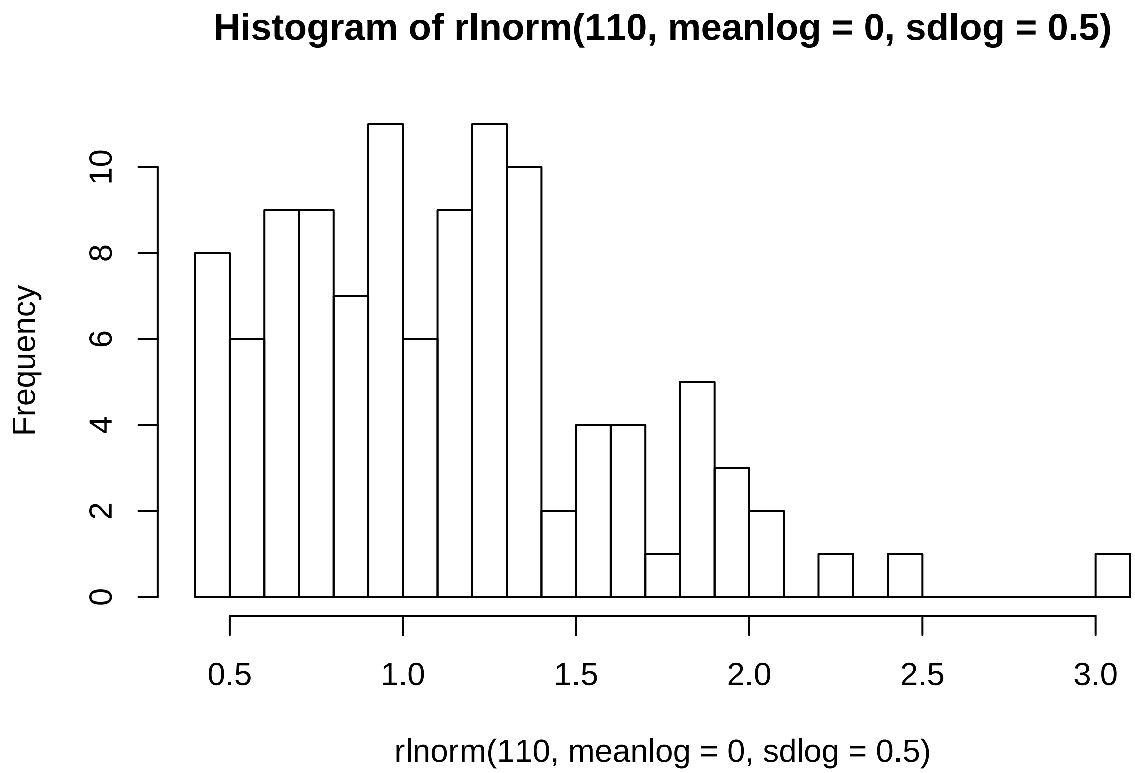
Assignment 2

1.

The following plot shows the histogram of price.



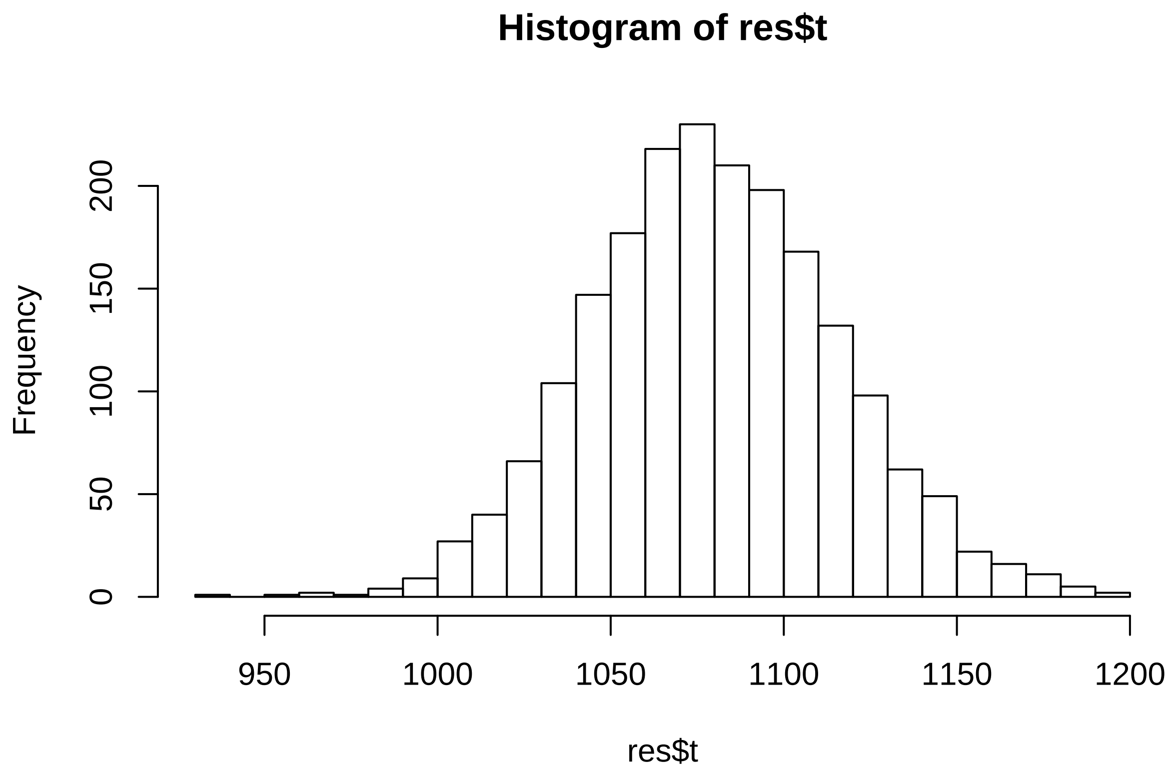
It can remind a lognormal distribution, for example as the following one:



The mean of the prices is 1080.4727273.

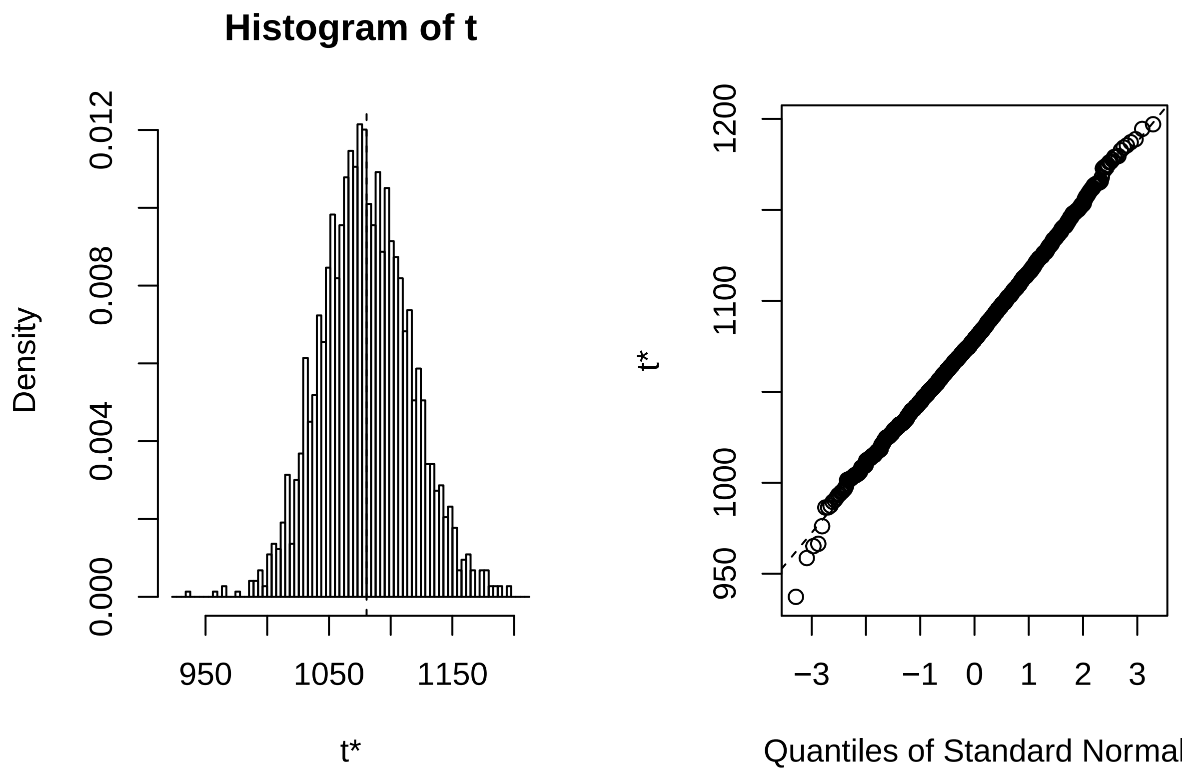
2.

The distribution of the mean of the prices with bootstrap is:



The bootstrap bias-correction is 1080.9628455, and the variance is 1284.358242.

Finally here are the plot for the bootstrap, and then the printings for the confidence intervals with three different methods (percentile, bca and first-order normal approximation):




```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 2000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = res, type = "perc")
##
## Intervals :
## Level     Percentile
## 95%   (1013, 1152 )
## Calculations and Intervals on Original Scale
```

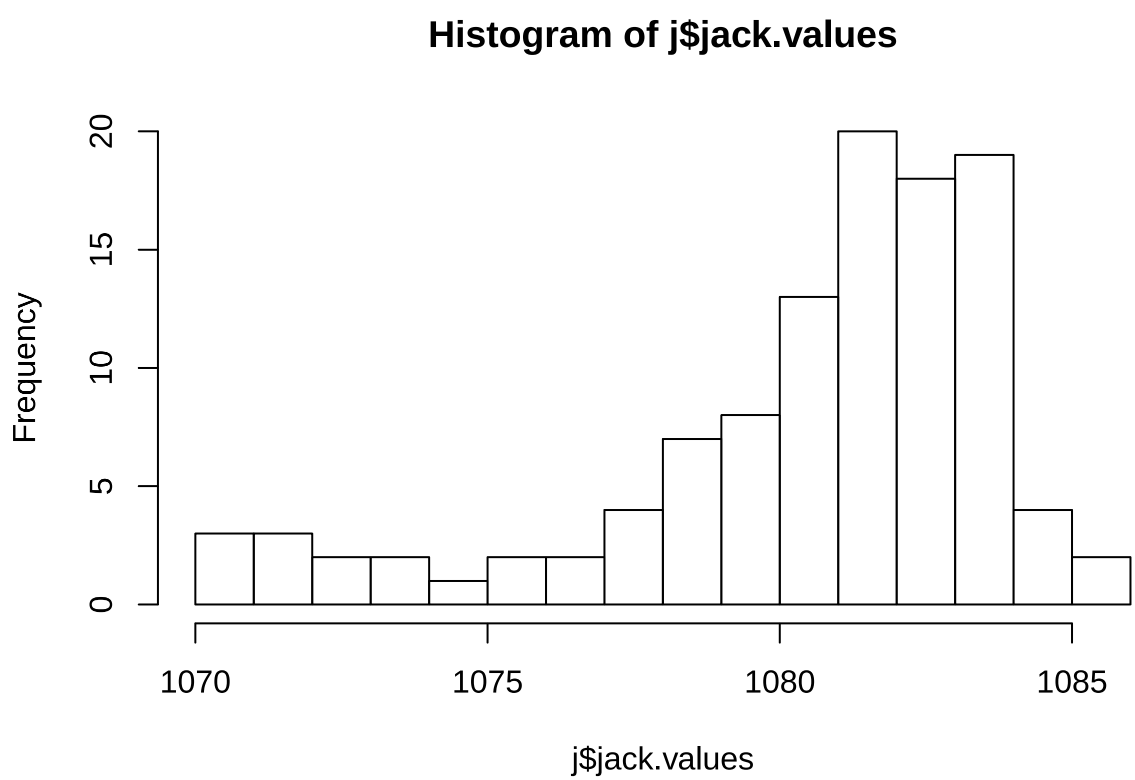
```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 2000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = res, type = "bca")
##
## Intervals :
## Level       BCa
## 95%   (1018, 1162 )
## Calculations and Intervals on Original Scale
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 2000 bootstrap replicates
##
```

```
## CALL :
## boot.ci(boot.out = res, type = "norm")
##
## Intervals :
## Level      Normal
## 95%   (1011, 1151 )
## Calculations and Intervals on Original Scale
```

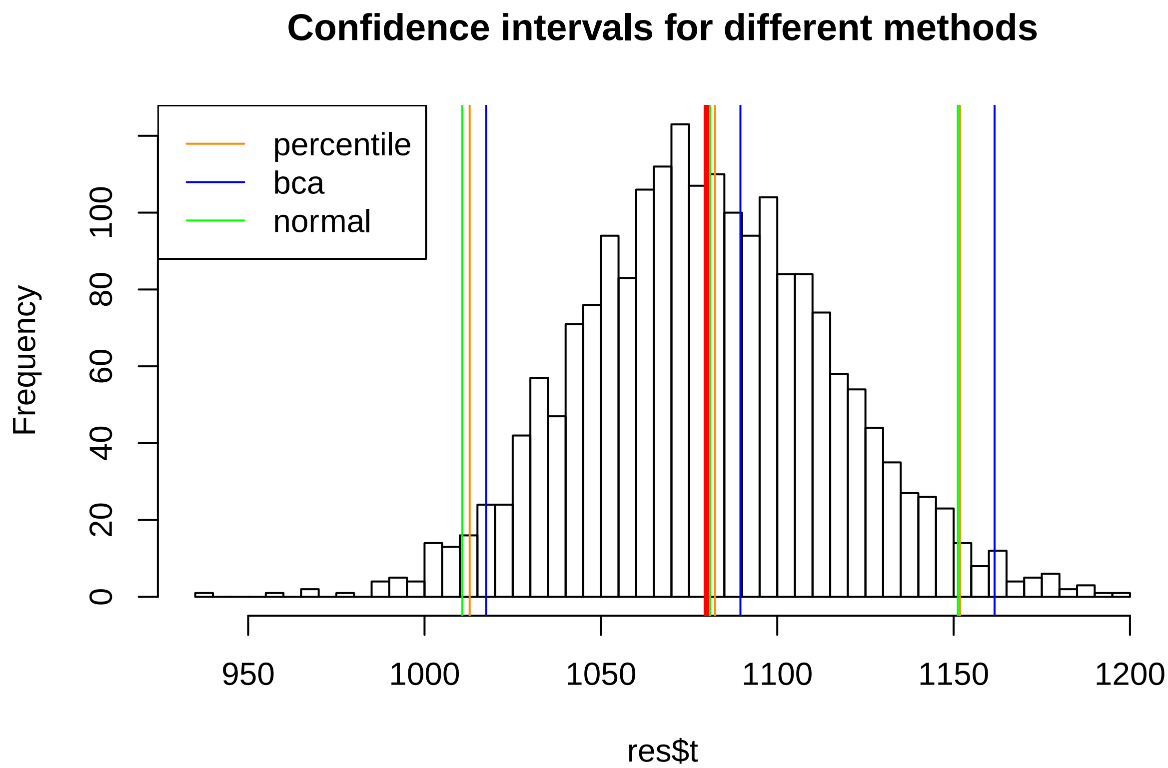
3.

Now let us try the Jackknife method. Here is the histogram of the means distribution:



The estimate of the variance is 1320.9110441. It is bigger than the estimate of the variance with bootstrap method, but the difference is not really high.

4.



We can see that the bca confidence interval is shifted compared to the other ones, and that its mean is also far from the actual mean. So we can already discard this confidence interval as the best. Then, we can see that the first-order normal approximation has a mean really close to the real one. But its confidence interval is a little bit larger than the one of percentile method. But I would rather the first-order normal approximation, which has its estimated mean almost equal to the real one.

Contribution

We both contributed half and half for the report, but we discussed everything together.

Appendix

```
## ---- echo = FALSE, message=FALSE, warning=FALSE------------------------
#1.1
library(xlsx)
data<-read.xlsx("C:/Users/Maxime/ENSAI/Liu/Computational statistics/Lab 5/lottery.xls", sheetIndex = 1,
X<-data$Day_of_year
Y<-data$Draft_No
lott<-data.frame(X=X,Y=Y)
n<-dim(lott)[1]
plot(X,Y)

## ---- echo = FALSE-------------------------------------------------------
plot(X,Y)
```

```
#1.2
m1<-loess(Y~X)
points(X,m1$fitted,col="red",type="l",lwd=3)

## ---- echo = FALSE-------------------------------------------------------
#1.3
T.fun<-function(X,Y){
  m<-loess(Y~X)
  Y_fit<-m$fitted
  Xb<-X[which.max(Y_fit)]
  Xa<-X[which.min(Y_fit)]
  t<-(max(Y_fit)-min(Y_fit))/(Xb-Xa)
  t
}
t<-T.fun(X,Y)
#Bootstrap:
B<-2000
boot.t<-c()
for (i in 1:B){
  perm<-sample(1:n,n,replace = TRUE)
  boot.t[i]<-T.fun(X[perm],Y[perm])
}
#Each element is a bootstraped t.
#we reject if 0 is not in this C.I
#sort(boot.t)[floor(2000*0.975)]
#sort(boot.t)[floor(2000*0.025)]
#p-value should be something like:
sum((boot.t)>0)/B
T.fun.boot<-function(data,n){
  data1<-data[n,]
  X<-data1$X
  Y<-data1$Y
  m<-loess(Y~X)
  Y_fit<-m$fitted
  Xb<-X[which.max(Y_fit)]
  Xa<-X[which.min(Y_fit)]
  t<-(max(Y_fit)-min(Y_fit))/(Xb-Xa)
  t
}
library(boot)
res<-boot(lott,T.fun.boot,2000)
# t.vect<-c()
# for (i in 1:2000){
#   t.vect[i]<-T.fun.boot(lott,n=sample(1:n,n,replace = TRUE))
# }
sum(res$t>0)/B

## ---- echo = FALSE-------------------------------------------------------
#1.4
perm.func<-function(data,B){
  t<-T.fun(data$X,data$Y)
  count<-0
  for(try in 1:B){
```

```r
    if(abs(T.fun(X[sample(1:n,n,replace=FALSE)],Y))>abs(t))
      {count<-count+1}
  }
  p<-count/B
  return(p)
}
cat("The p-value for perm.test is:",perm.func(lott,2000))

## ---- echo = FALSE--------------------------------------------------------
#1.5
Y.fun<-function(alpha,x,hyperparam=list(mean,sd)){
  new.x<-c()
  for (i in 1:length(x)){
    new.x[i]<-max(0,min(alpha*x[i]+rnorm(1,hyperparam$mean,hyperparam$sd),366))
  }
  return(new.x)
}
p_values<-c()
for (alpha in seq(0.1,10,0.1)){
  new.Y<-Y.fun(alpha=alpha,x=X,hyperparam = list(mean=183,sd=10))
  p_values<-c(p_values,perm.func(data.frame(X=1:366,Y=new.Y),B=200))
}
plot(seq(0.1,10,0.1),p_values,type="b",col="red",pch=19)
lines(-2:12,rep(0.05,15),lwd=3)

## ---- echo = FALSE--------------------------------------------------------
pri <- read.xlsx("C:/Users/Maxime/ENSAI/Liu/Computational statistics/Lab 5/prices1.xls", sheetIndex = 1

# 1.
hist(pri$Price, 20)

## ---- echo = FALSE--------------------------------------------------------
hist(rlnorm(110, meanlog = 0, sdlog = 0.5), 20)

## ---- echo = FALSE--------------------------------------------------------
stat1<-function(data,n){
  data1=data[n,];
  return(mean(data1$Price))
}
set.seed(12345)
res <- boot(pri,stat1,R=2000)
hist(res$t,20)

v <- 1/1999*sum((res$t-mean(res$t))^2)
bc <- 2*mean(pri$Price) - mean(res$t)

## ---- echo = FALSE--------------------------------------------------------
perc <- boot.ci(res, type = "perc")
bca <- boot.ci(res, type = "bca")
nor <- boot.ci(res, type = "norm")

plot(res)
print(perc)
```

```
print(bca)
print(nor)

## ---- echo = FALSE, warning = FALSE-------------------------------------
library(bootstrap)
stat1 <- function(indice, data){
  mean(data$Price[indice])
}
j <- jackknife(1:110, stat1, data=pri)
hist(j$jack.values, 20)

## ---- echo = FALSE-------------------------------------------------------
hist(res$t,40, main = "Confidence intervals for different methods")
abline(v = mean(res$t), col = "red", lwd = 3)
abline(v = perc$percent[c(4,5)], col = "darkorange")
abline(v = bca$bca[c(4,5)], col = "blue")
abline(v = nor$normal[c(2,3)], col = "green")
abline(v = mean(perc$percent[c(4,5)]), col = "darkorange")
abline(v = mean(bca$bca[c(4,5)]), col = "blue")
abline(v = mean(nor$normal[c(2,3)]), col = "green")
legend(x = "topleft", legend = c("percentile", "bca", "normal"), lty = 1,
       col = c("darkorange", "blue", "green"))

## ----code=readLines(knitr::purl("C:/Users/Maxime/ENSAI/Liu/Computational statistics/Lab 5/Group 3-lab
##
```